

Professora: paula prata

rEALIZAÇÃO POR âNGELO MORGADO, 43855

Bruno monteiro, 43994

dUARTE ARRIBAS, 44585

hENRIQUE JESUS, 43931

RELATÓRIO

1º semestre

Ano Letivo de 2020/2021

Programação Orientada a Objetos

**Índice**

[Introdução 1](#_Toc59289781)

[Correndo a aplicação 2](#_Toc59289782)

[Userlist 4](#_Toc59289783)

[Password Admin principal 4](#_Toc59289784)

[Exemplos de utilizadores já registados 4](#_Toc59289785)

[Classes 5](#_Toc59289786)

[User 5](#_Toc59289787)

[Client 5](#_Toc59289788)

[Admin 5](#_Toc59289789)

[Login 5](#_Toc59289790)

[Register 5](#_Toc59289791)

[Order 5](#_Toc59289792)

[Product 6](#_Toc59289793)

[NationalProduct e InternationalProduct 6](#_Toc59289794)

[Estatística 6](#_Toc59289795)

[ProfitList 7](#_Toc59289796)

[Exceptions 7](#_Toc59289797)

[AlreadyRegisteredException 7](#_Toc59289798)

[OptionNotInRangeException 7](#_Toc59289799)

[Input 7](#_Toc59289800)

[WriteToFile 7](#_Toc59289801)

[Menu 7](#_Toc59289802)

[Option 7](#_Toc59289803)

[Main 7](#_Toc59289804)

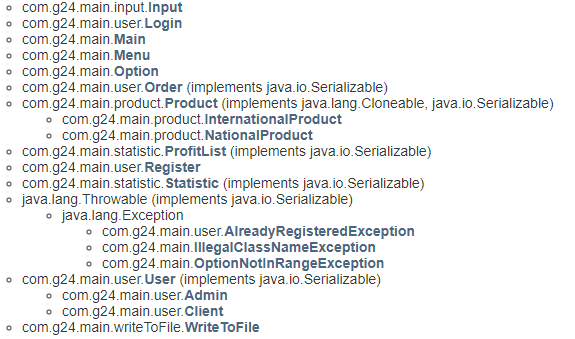
[Conclusão 8](#_Toc59289805)

# Introdução

Uma agência de viagens tem-se como uma empresa, que se destina, não só, a ser intermediaria entre os prestadores de serviços turísticos e os seus utilizadores. Sendo esta online faz, também, a gestão dos seus clientes e possíveis compras, através da verificação de inputs e, posterior, envio de feedback sobre estas.

Para a resolução deste problema, estabelecemos um mapa com base na vasta gama de aplicações de compra online, que nos ajuda a organizar a nossa aplicação com as mais comuns praticas de programação e user experience, descartando as mais inusitadas e, possivelmente, prejudiciais a navegação da mesma.

A nossa aplicação esta estruturada da seguinte forma:



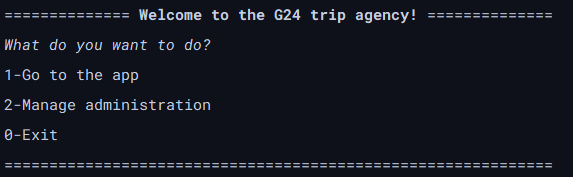
A main package: com.g24.main alberga toda a aplicação, sendo que, ulteriormente, se subdividira nas packages:

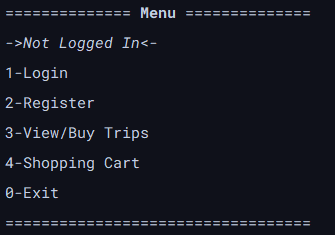
* com.g24.main.input: A package que contém a class Input, que reúne todos os métodos estáticos de pedida de input ao utilizador e as suas verificações.
* com.g24.main.writeToFile: A pacakge que contém a classe WriteToFile, que se destina a guardar as diversas informações do programa em ficheiros.
* com.g24.main.user: A package que contém todas as classes inerentes aos utilizadores (sejam estes, clientes ou admins) e as suas funções (login, register, entre outros); possui ainda classes de gestão dos pedidos dos clientes.
* com.g24.main.product: A package que contém as classes que definem cada produto e as suas propriedades.
* com.g24.main.statistic: A package que contém as classes de produção de documentos estatísticos, alusivos aos diferentes pedidos entre utilizador e respetivos produtos.

De modo a profissionalizar a aplicação, geramos documentação javadocs para classes, métodos e variáveis publicas e protected, de modo, a coordenar o trabalho entre os elementos do grupo.

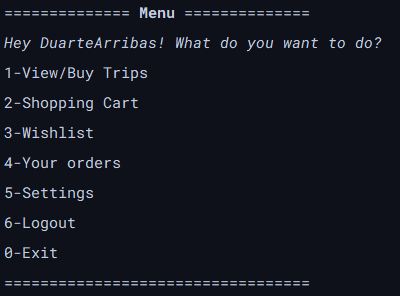
# Correndo a aplicação

Ao correr a aplicação, um menu de *Welcome* é inicializado e o utilizador tem a opção de registar um novo admin (caso saiba a *password* utilizada para aceder a esse menu; caso a erre, o programa encerra, de modo a evitar algoritmos de *bruteforce*), de entrar para a aplicação ou de a encerrar.

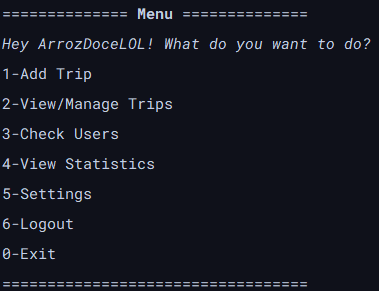


Caso entre na aplicação, um menu para utilizadores não autenticados é apresentado, onde o utilizador poderá fazer o login (autenticar-se), registar-se, ver as diferentes viagens (podendo adicionar ao carrinho), ver o carrinho de compras (compras estas, que só poderão ser confirmadas após a autenticação do utilizador) e sair do programa. 

Ao fazer o login, a aplicação verifica que tipo de utilizador as credenciais correspondem.

Se for cliente, diferentes opções ficam disponíveis, tais como a possibilidade de adicionar à wishlist os produtos e de a ver, confirmar as suas compras, ver os seus pedidos, atualizar os dados da conta, incluindo o facto de poder comprar uma membership (tornando-se um utilizador VIP, que terá descontos na compra de viagens), logout e saída do programa.

Se for admin, outras opções ficam disponíveis, tais como a adição de novas viagens e modificação das existentes, verificação da informação dos utilizadores, observação das estatísticas geradas pela compra dos produtos da agência, atualizar os dados da conta, logout e saída do programa.



Todas estas opções do programa foram resumidamente explicadas e devem ser vistas em ação.

# Userlist

Lista de utilizadores já registados na aplicação para fins de teste da mesma:

## Password Admin principal

adminG24\_oop

Esta password serve para criar novos utilizadores com permissões de administrador no menu . Apenas deve ser sabida por utilizadores admin.

## Exemplos de utilizadores já registados

# Classes

## User

Tem-se como a superclasse de todos os utilizadores, onde define as propriedades gerais destes, os seus getters e setters. Possui métodos estáticos para apresentar diversas caraterísticas dos utilizadores e para modificar as já definidas***.***

Esta classe não deverá ser instanciada.

## Client

Sendo esta subclasse do user, especifica-o, de modo a obter caraterísticas, que são inerentes apenas ao cliente, como modificações a este, gestão do carrinho e gestão da wishlist.

### Admin

Tal como a classe de cima, especifica o cliente. Não dispondo de um vasto conjunto de métodos próprios, tem como objetivo de ser utilizada em comparações por uso do instanceof.

## Login

Ocupa-se da autenticação dos utilizadores, isto é, compara os dados introduzidos com os do ficheiro, instanciando o utilizador, caso os dados estejam corretos.

## Register

Ocupa-se do registo de novos utilizadores na aplicação, dispondo de um vasto conjunto de métodos de verificação de dados e erros, de modo a evitar dados mal introduzidos ou possível duplicação.

## Order

A classe Order é referente aos pedidos de cada cliente, contendo esta:

* O ID do pedido, sendo este gerado de forma sequencial, isto é, cada vez que um objeto do tipo order é instanciado este é incrementado.
* Uma ArrayList de produtos, esta armazenará todos os produtos adquiridos num certo pedido pelo cliente.
* Duas variáveis do tipo string que guardam a data e hora, respetivamente, de quando o pedido foi processado.
* O preço (float) total dos produtos que este pedido contém, ou seja, se o pedido tiver o Produto\_A (100€) e Produto\_B (200€) então o preço do pedido será a soma dos preços de cada produto, portanto 300€.
* O dispêndio (float) que o pedido tem para a empresa, esta variável serve para facilitar o cálculo estatístico do lucro.

De facto, quando um cliente faz um pedido é instanciado um objeto do tipo Order, no seu construtor o método addOrder() é chamado, neste será definido a data e hora através do método getTimeHour(), o preço e o dispêndio pelo setPrice() e setSpend(), respetivamente e os produtos adquiridos pelo cliente serão guardados num ArrayList de produtos.

## Product

A classe product é uma classe cuja existência é contabilizar todos os produtos existentes e as suas propriedades.

Primeiramente, o nosso produto contém propriedades inerentes a este, sendo estas a designação, a duração da viagem, o preço pago pela empresa, a comissão da empresa, e o preço do utilizador, que vai ser o preço da empresa mais a comissão que servirá como lucro e o local de partida.

### NationalProduct e InternationalProduct

Além de uma superclasse de productos, nós decidimos dividir as viagens em duas subclasses, surgindo assim as viagens nacionais, associadas à subclasse 'nationalProduct' e as viagens, associadas à subclasse 'internationalProduct',desta forma, será possível fazer as estatísticas das viagens nacionais e internacionais em separado.

Ao entrar no menu o cliente, autenticado ou não, poderá aceder a uma lista que contém as viagens todas expostas. Ao clicar numa viagem à sua escolha, o cliente terá à sua disposição todas as informações associadas ao produto desejado, apresentadas de forma apelativa e fácil de entender. Após isso serão apresentadas opções ao utilizador diferentes dependendo do seu estado. Caso o seja um cliente não autenticado ele apenas poderá adicionar a viagem ao seu carrinho de compras ou voltar atrás à lista das viagens, caso seja um cliente autenticado, este poderá fazer tudo o que o anterior fazia mais adicionar o produto à sua lista de desejos. Se o utilizador for um admin, este terá à sua disposição a opção de adicionar uma viagem nova no menu principal e na lista de produtos, poderá alterar a viagem ou simplesmente removê-la.

Todas estas ações descritas acima são métodos da classe produtos.

## Estatística

Classe que inclui funções para calcular diferentes dados estatísticos para a nossa agência. Consideramos essencial a criação de métodos que nos permitisse calcular o lucro obtido em cada viagem, a média de viagens nacionais e internacionais adquiridas pelos clientes, a viagem mais vendida, a viagem mais cara e a mais barata e, por fim, o lucro médio obtido, tendo em conta os pedidos realizados.

## ProfitList

Classe que indica a designação e lucro de uma viagem. Esta irá ser útil para mais tarde produzirmos uma arraylist com designação e lucro de cada viagem que a agência disponibiliza aos seus clientes. Esta arraylist será fundamental para a produção de alguns métodos estatísticos

## Exceptions

Para a verificação de certos erros, criámos certas classes de exceções para a sua gestão.

### AlreadyRegisteredException

Utilizada para quando o utilizador que será registado já estiver no ficheiro. ***IllegalClassNameException***

Utilizada para quando a classe que será utilizada para saber o tipo de utilizador não é um tipo de utilizador.

### OptionNotInRangeException

Utilizada para quando a opção escolhida pelo utilizador está fora do range de opções que tem à sua disposição.

## Input

Classe inspirada na classe Ler, desenvolvida na disciplina, que possui todas as possíveis verificações de dados para todos os tipos primitivos.

## WriteToFile

Classe para guardar as diversas informações da aplicação em ficheiros, para que mantenha consistência antes e depois de a fechar.

## Menu

Destinada a mostrar menus, gere ainda, alguma escolha de opções.

## Option

Auxiliada pela classe definida em cima, gere a escolha de opções, dispondo o utilizador das diversas opções que pode escolher.

## Main

A classe executada de início. Possui a leitura dos ficheiros para a aplicação e contém o mapa que delimita as opções que o utilizador poderá escolher na aplicação.

# Conclusão

Em suma, este trabalho trouxe um vasto conhecimento da aplicação, dos conhecimentos de programação orientada a objetos e organização entre os membros do grupo. Aprendemos a utilizar ferramentas inerentes a linguagem de programação java, aplicando, pois, os conhecimentos já aprendidos.

De qualquer maneira, pretendemos continuar a aprender, de modo a aplicar estes conhecimentos a aplicações futuras e possíveis aplicações em contexto profissional.